Solving a Scheduling Problem for a Quantum Computing Architecture Using Constraint Programming

Jean-Paul Watson[*], Cynthia A. Phillips[*], and Robert D. Carr[*]

The last decade has witnessed remarkable growth in research involving quantum computing architectures. This surge of interest is due to the theoretical abilities of quantum architectures to significantly accelerate – relative to classical computers – solution of broad classes of important computational problems, including factoring, search, and simulation of quantum chemistry. Sandia is investigating quantum architectures in support of the development of advanced computing architectures, in addition to advancing understanding of quantum effects in nano-scale engineered systems. Current Sandia quantum efforts are specifically focused on building small-scale architectures to explicitly test current understanding of quantum mechanics.

In "conventional" (i.e., non-adiabatic) proposed quantum computing architectures, such as the bilinear array architecture introduced by Hollenberg et al., a single logical quantum bit or qubit will be implemented using a network of physical qubits. The additional physical qubits serve to implement the extensive error checking required to obtain correct behavior of the logical qubit. The efficiency of error correction will fundamentally influence the performance of future quantum computers, in terms of latency/speed and error threshold.

Execution of quantum error correction requires the scheduling of both error-correcting operations and the movement of qubits – or the information associated with qubits – to locations where they can interact in gates. This general problem is an instance of an multi-processor, precedence-constrained scheduling problem, with additional and unique routing and other architecture-specific constraints. The optimization objective is to minimize schedule makespan, reductions in which translate into higher-performing quantum architectures.

We will describe the quantum architecture scheduling problem in detail and discuss the formulation and solution of this problem using constraint programming. Constraint programming is a relatively new optimization paradigm, with some prior demonstrated success in solving large-scale scheduling and routing problems. In contrast to mixed-integer programming, search in constraint programming is based on logical inference and variable-domain pruning. We will compare results with those obtained by mixed-integer programming, and discuss the complementary strengths of the two technologies.